TÉCNICO LISBOA

Deep Learning

MEEC, MEIC, MECD

Second Exam

February 2, 2024

Version A

Instructions

- You have 120 minutes to complete the exam.
- Make sure that your test has a total of 10 pages, then write your full name and student number on this page (and your number in all the other pages).
- The test has a total of 17 questions, with a maximum score of 100 points. The questions have different levels of difficulty. The point value of each question is provided next to the question number.
- Please provide your answer in the space below each question. If you make a mess, clearly indicate your answer.
- The exam is open book and open notes. You may use a calculator, but any other type of electronic or communication equipment is not allowed.
- Good luck.

Part 1	**Part 2**	**Part 3, Pr. 1**	**Part 3, Pr. 2**	**Total**
32 points	18 points	25 points	25 points	100 points

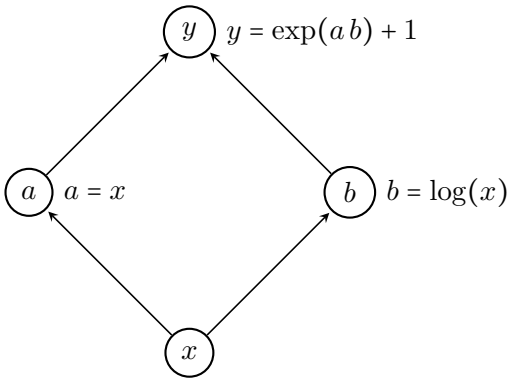
Part 1: Multiple Choice Questions (32 points)

In each of the following questions, indicate your answer by *checking a single option.*

1. (4 points) When handling a classification task with 10 labels, what is the standard choice for the final layer of a deep network?
   - ☐ A fully connected linear layer.
   - ☐ A ReLU activation.
   - ☐ A hyperbolic tangent activation.
   - ■ **None of the above is suitable.**

2. (4 points) What property does a max-pooling layer exhibit?
   - ☐ Exact shift invariance (holds for any shift).
   - ■ **Approximate shift invariance (holds for some, but not all, shifts).**
   - ☐ Exact shift equivariance (holds for any shift).
   - ☐ Approximate shift equivariance (holds for some, but not all, shifts).

3. (4 points) Consider the following computation graph, where $x > 0$ is real positive variable:



   Which of the following functions is $\frac{dy}{dx}$?
   - ☐ $x^x \log(x)$.
   - ☐ $x^x \log(x) + 1$.
   - ■ $x^x(\log(x) + 1)$**.**
   - ☐ None of the previous answers.

   Solution:

$$\frac{dy}{dx} = \underbrace{\frac{\partial y}{\partial a}}_{b\exp(ab)} \underbrace{\frac{da}{dx}}_{1} + \underbrace{\frac{\partial y}{\partial b}}_{a\exp(ab)} \underbrace{\frac{db}{dx}}_{1/x} = \exp(x\,\log(x))(\log(x)+1) = \underbrace{\exp(\log(x^x))}_{x^x}(\log(x)+1).$$

4. (4 points) Which of the following statements about autoregressive RNN-based language models is **false**?
   - ■ **RNNs make a Markov assumption, hence they can only remember the most recent n words they generate, for some $n \in \mathbb{N}$.**

- ☐ RNNs can handle sequences of variable length by maintaining a hidden state that captures information from previous elements in the sequence.
- ☐ RNNs have unbounded memory, however, for long sequences they have a tendency to "remember" less accurately the initial words they have generated.
- ☐ Learning RNNs by gradient descent suffers from the vanishing gradients problem.

5. (4 points) Let $\boldsymbol{s} = [0.25, 0.25, 0.25 + a, 0.25 - a]^T$, where $a \in [-0.25, 0.25]$ is a real number, be a vector of logits. A probability vector $\boldsymbol{p} = [p_1, \ldots, p_4]^T$ is obtained by computing the softmax transformation of $\boldsymbol{s}$. Then,

   - ☐ p_1 is a strictly increasing function of a.
   - ☐ p_1 is a strictly decreasing function of a.
   - ■ **p_1 is a non-monotonic (and non-constant) function of a.**
   - ☐ p_1 does not depend on a.

   Solution: We have $p_1 = e^{0.25}/\left(e^{0.25}(2 + e^a + e^{-a})\right)$, which is easy to see that is symmetric around $a = 0$, thus the first two options are wrong, and is not a constant (because $e^a + e^{-a}$ is not a constant), thus the fourth option is wrong.

6. (4 points) How do *bidirectional LSTMs* (BiLSTMs) enhance the model's performance for tasks involving sequential data, in comparison with unidirectional LSTMs?

   - ☐ BiLSTMs increase the training speed by processing data in parallel, thereby reducing the need for long training periods typical of unidirectional LSTMs.
   - ☐ BiLSTMs utilize a simpler gating mechanism than unidirectional LSTMs, which reduces the model's complexity and improves computational efficiency.
   - ■ **BiLSTMs can capture dependencies in both forward and backward directions of the sequence, providing the model with a more comprehensive understanding of the context.**
   - ☐ BiLSTMs improve the model's ability to handle non-sequential data by incorporating external memory components, unlike unidirectional LSTMs.

7. (4 points) How does the attention mechanism differ fundamentally from the approach used in recursive models like RNNs and LSTMs in processing sequential data?

   - ☐ The attention mechanism uses convolutional layers to process sequential data, whereas RNNs and LSTMs use fully connected layers for this purpose.
   - ■ **Unlike RNNs and LSTMs, the attention mechanism allows for parallel processing of sequential data, enabling the model to focus on different parts of the sequence simultaneously.**
   - ☐ The attention mechanism requires significantly more training data than RNNs and LSTMs to achieve comparable performance on sequential tasks.
   - ☐ RNNs and LSTMs can only process numerical data sequences, while the attention mechanism is capable of processing both numerical and non-numerical data.

8. (4 points) We observe that a machine translation model trained on an English to Portuguese dataset consistently translates mentions of doctors as masculine ("the doctor" is always translated to "o doutor" ou "o médico") even in cases where there are indications (pronouns, names) that the correct gender is feminine. What could be done to change this behaviour?

   - ☐ Retrain the model for fewer epochs, because this is a symptom of overfitting.
   - ☐ Retrain the model masking all mentions of "doctor" in the dataset.
   - ■ **Retrain the model after data augmentation where for each instance that contains a mention of "doctor" that is male, the same sentence is added with a mention of "doctor" that is female.**
   - ☐ Retrain the model after data augmentation where for each instance that contains a mention of "doctor" that is female, the same sentence is added with a mention of "doctor" that is male.

Part 2: Short Answer Questions (18 points)

Please provide **brief** answers (1-2 sentences) to the following questions.

1. (6 points) Explain how a bidirectional RNN works and what advantages it has over the unidirectional version.

   Solution: Bidirectional RNNs combine a left-to-right RNN with a right-to-left RNN. After propagating the states in both directions, the states of the two RNNs are concatenated. Bidirectional RNNs are used as sequence encoders, their main advantage is that each state contains contextual information coming from both sides. Unlike unidirectional RNNs, they have the same focus on the beginning and on the end of the input sequence.

2. (6 points) Explain what is the vanishing gradient problem in RNNs training and provide an example of one technique that aims at mitigating it.

   Solution: The vanishing gradient problem occurs when the gradients needed for backpropagation become too small leading to small or no weight updates. In RNNs we have backpropagation through time and inputs in the beginning of a longer sequence are greatly impacted by the vanishing gradient since they are multiplied by the low value weight matrices more times. LSTM and GRU cells were designed to mitigate this problem.

3. (6 points) How are the outputs of multi-attention heads aggregated in the transformers architecture? Name one advantage of having multi-head attention rather than a single-head one.

   Solution: The outputs of multi-head attention are concatenated through a feed-forward neural network whose weights are learned during training. Multi-attention heads allow for attending to parts of the sequence differently (e.g. longer-term dependencies versus shorter-term dependencies), the network then learns also the weights to attribute to each attention output, thus encoding more information than what a single attention head would provide.

Part 3: Problems (50 points)

Problem 1. Time series prediction (25 points)

Consider a time series $\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_t, ...$, with $\boldsymbol{u}_t \in \mathbb{R}^3$ and $t \in \mathbb{N}$, which verifies

$$\boldsymbol{u}_{t+1} = 5 \max(2\boldsymbol{u}_t - 3\boldsymbol{u}_{t-1} - \boldsymbol{u}_{t-2} + 1, 0),$$

and we assume $\boldsymbol{u}_t = 0$ for $t < 0$.

1. (10 points) Suppose that we want to build a predictor for the time series using a neural network. In particular, we want to build a neural network that takes as input $\boldsymbol{x} = [\boldsymbol{u}_{t-2}^\top, \boldsymbol{u}_{t-1}^\top, \boldsymbol{u}_t^\top]^\top$ and returns as output $\boldsymbol{y} = \boldsymbol{u}_{t+1}$. Show that the dynamics of the time series can be represented exactly using a neural network with a single hidden layer. In particular, indicate the dimension, weights, and activation function of both hidden and output layers.

Solution: The time series can be represented using a neural network with a single hidden layer with 3 units and ReLU activation, where

$$\boldsymbol{W}_1 = \begin{bmatrix} -1 & 0 & 0 & -3 & 0 & 0 & 2 & 0 & 0 \\ 0 & -1 & 0 & 0 & -3 & 0 & 0 & 2 & 0 \\ 0 & 0 & -1 & 0 & 0 & -3 & 0 & 0 & 2 \end{bmatrix}, \qquad \boldsymbol{b}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

The output of the hidden layer is thus a vector $\boldsymbol{h} \in \mathbb{R}^3$ with

$$\begin{aligned} \boldsymbol{h}(\boldsymbol{x}) &= \mathrm{ReLU}(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1) \\ &= \mathrm{ReLU}\left(\begin{bmatrix} 2u_{t,1} - 3u_{t-1,1} - u_{t-2,1} + 1 \\ 2u_{t,2} - 3u_{t-1,2} - u_{t-2,2} + 1 \\ 2u_{t,3} - 3u_{t-1,3} - u_{t-2,3} + 1 \end{bmatrix} \right). \end{aligned}$$

The output layer consists of 3 linear units, and

$$\boldsymbol{W}_o = 5\,\mathbf{I}_{3\times 3}, \qquad \boldsymbol{b}_o = \mathbf{0}.$$

The output thus comes $\boldsymbol{y} = 5\,\boldsymbol{h}(\boldsymbol{x})$.

2. (7 points) Suppose that we observe the sequence $\{\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T\}$ for some large T. Indicate how the sequence can be used to build a dataset $\{(\boldsymbol{x}_n, \boldsymbol{y}_n), n = 1, \ldots, N\}$ that can be used to learn the predictor in 1. Specifically, indicate what the input $\boldsymbol{x}_n$ and output $\boldsymbol{y}_n$ should be in each of these pairs. For any given T, how many samples N would you get in your dataset?

   Solution: Since the goal of the network is to predict $\boldsymbol{u}_{t+1}$ as a function of $\boldsymbol{u}_t$, $\boldsymbol{u}_{t-1}$, and $\boldsymbol{u}_{t-2}$, we can build a dataset consisting of pairs $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ where $\boldsymbol{x}_n = [\boldsymbol{u}_{t-2}^\top, \boldsymbol{u}_{t-1}^\top, \boldsymbol{u}_t^\top]^\top$ and $\boldsymbol{y}_n = \boldsymbol{x}_{t+1}$ for $t \in \{2, \ldots, T-1\}$. The sequence $\{\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T\}$ would thus yield a total of $T-2$ samples.

3. (8 points) Suppose that we want to train the network to learn the predictor in 1, for which we now have a dataset with N input-output pairs. Further suppose that we want to use ℓ_2 regularization. Indicate the loss function that you would use to train the network, including the regularization term. Identify one possible reason for including ℓ_2 regularization in the loss function.

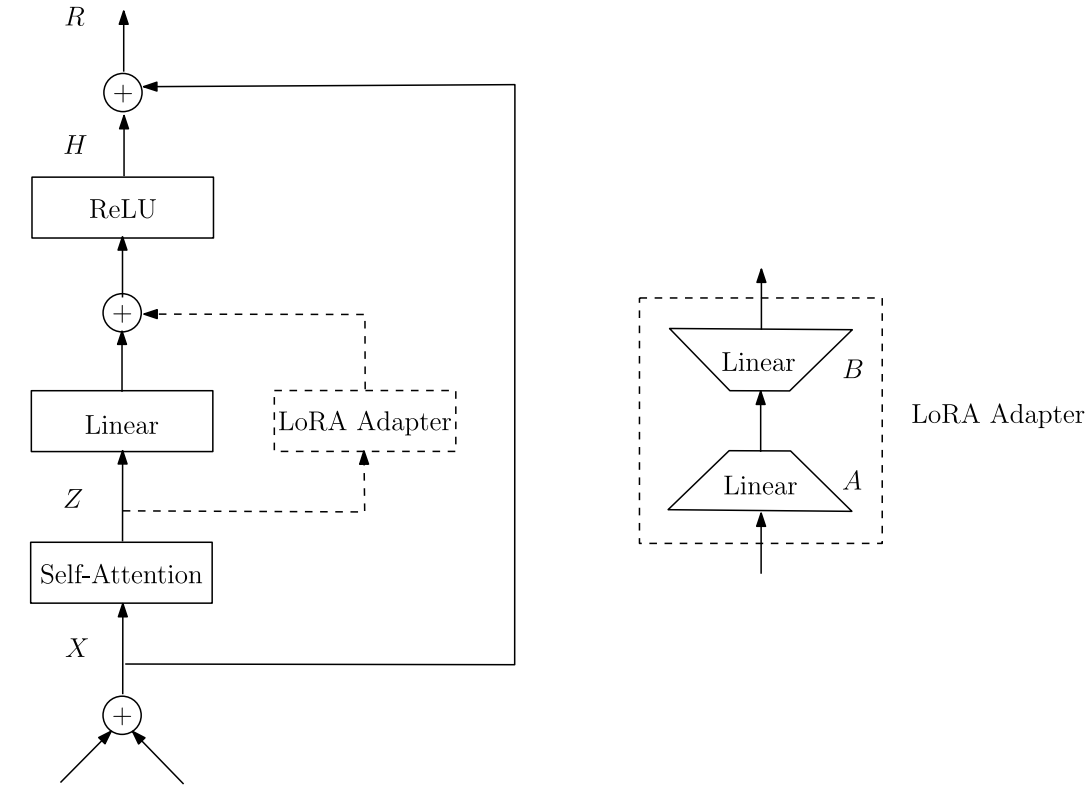
   Solution: Since the goal of the network is to predict $\boldsymbol{u}_{t+1}$ as a function of $\boldsymbol{u}_t$, $\boldsymbol{u}_{t-1}$, and $\boldsymbol{u}_{t-2}$, we are dealing with a regression problem for which the squared error is a convenient loss function. As such, the loss function would be:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2N} \sum_{n=1}^{N} \|\boldsymbol{y}_n - f(\boldsymbol{x}_n; \boldsymbol{\theta})\|_2^2 + \frac{\lambda}{2} \|\boldsymbol{\theta}\|_2^2,$$

   where $\boldsymbol{\theta}$ represents the parameters of the network, $f(\boldsymbol{x}_n; \boldsymbol{\theta})$ represents the output of the network for input $\boldsymbol{x}_n$, and each $(\boldsymbol{x}_n, \boldsymbol{y}_n)$ is an input-output pair in the dataset. The use of regularization mechanisms such as ℓ_2 regularization is one way to mitigate overfitting.

Problem 2: Transformers and Adapters (25 points)

Consider the following transformer block with an optional LoRA adapter placed in parallel, and a skip connection. The linear, ReLU, and LoRA adapter are applied elementwise to each token in the sequence. The LoRA adapter consists of a linear "down-project" followed by a linear "up-project" operation.



The input sentence is "Adapters are great" and we have the following word embeddings:

$$\boldsymbol{w}_{\text{Adapters}} = [-2, 1, 0, 0]^\top, \quad \boldsymbol{w}_{\text{are}} = [0, -1, 0, 0]^\top, \quad \boldsymbol{w}_{\text{great}} = [0, 0, -1, 1]^\top.$$

The positional encodings are

$$\boldsymbol{p}_1 = [1, 0, 0, 0]^\top, \quad \boldsymbol{p}_2 = [0, 1, 0, 0]^\top, \quad \boldsymbol{p}_3 = [0, 0, 1, 0]^\top.$$

The transformer uses multi-head attention with two attention heads, with scaled dot-product attention. The first attention head has projection matrices

$$\boldsymbol{W}_Q^1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad \boldsymbol{W}_K^1 = \boldsymbol{W}_V^1 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \\ -2 & 0 \\ 0 & -2 \end{bmatrix},$$

and the second attention head has $\boldsymbol{W}_Q^2 = -\boldsymbol{W}_Q^1$, $\boldsymbol{W}_K^2 = -\boldsymbol{W}_K^1$, $\boldsymbol{W}_V^2 = -\boldsymbol{W}_V^1$. The output projection matrix is the identity, $\boldsymbol{W}_O = \boldsymbol{I}$.

The linear layer after the self-attention has zero bias vector and the following weight matrix:

$$\boldsymbol{W}_{FF} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

The adaptor uses matrices $\boldsymbol{A} = \boldsymbol{a}^\top$ and $\boldsymbol{B} = \boldsymbol{a}$ with $\boldsymbol{a} = [1, 1, 1, 1]^\top$, and no bias.

1. (10 points) Compute the output of the network **without** the adapter.

Solution: The input representations after adding the positional encodings is

$$\boldsymbol{X} = \begin{bmatrix} -2 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The query, key, and value matrices for the first attention head are

$$\boldsymbol{Q}^1 = \boldsymbol{X}\boldsymbol{W}_Q^1 = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & -1 \end{bmatrix},$$

$$\boldsymbol{K}^1 = \boldsymbol{V}^1 = \boldsymbol{X}\boldsymbol{W}_K^1 = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 2 \\ -2 & 0 \\ 0 & -2 \end{bmatrix} = \begin{bmatrix} -2 & 2 \\ 0 & 0 \\ 0 & -2 \end{bmatrix}.$$

The attention probability matrix is

$$\boldsymbol{P}^1 = \text{Softmax}\left(\frac{1}{\sqrt{2}}\boldsymbol{Q}^1(\boldsymbol{K}^1)^\top\right) = \text{Softmax}\left(\frac{1}{\sqrt{2}} \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} -2 & 2 \\ 0 & 0 \\ 0 & -2 \end{bmatrix}^\top\right)$$

$$= \text{Softmax}\left(\frac{1}{\sqrt{2}} \begin{bmatrix} 4 & 0 & -2 \\ 0 & 0 & 0 \\ -2 & 0 & 2 \end{bmatrix}\right) = \begin{bmatrix} 0.932 & 0.055 & 0.013 \\ 0.333 & 0.333 & 0.333 \\ 0.045 & 0.187 & 0.768 \end{bmatrix}$$

which leads to

$$\boldsymbol{Z}^1 = \boldsymbol{P}^1\boldsymbol{V}^1 = \begin{bmatrix} 0.932 & 0.055 & 0.013 \\ 0.333 & 0.333 & 0.333 \\ 0.045 & 0.187 & 0.768 \end{bmatrix} \begin{bmatrix} -2 & 2 \\ 0 & 0 \\ 0 & -2 \end{bmatrix} = \begin{bmatrix} -1.863 & 1.836 \\ -0.667 & 0 \\ -0.091 & -1.445 \end{bmatrix}.$$

For the second attention head, we have $\boldsymbol{Q}^2 = -\boldsymbol{Q}^1$ and $\boldsymbol{K}^2 = -\boldsymbol{K}^1$, hence $\boldsymbol{P}^1 = \boldsymbol{P}^2$. Therefore, $\boldsymbol{Z}^2 = \boldsymbol{P}^2\boldsymbol{V}^2 = -\boldsymbol{P}^1\boldsymbol{V}^1 = -\boldsymbol{Z}^1$. The output of the self-attention layer is therefore

$$\boldsymbol{Z} = [\boldsymbol{Z}^1, \boldsymbol{Z}^2] \underbrace{\boldsymbol{W}_O}_{\text{identity}} = \begin{bmatrix} -1.863 & 1.836 & 1.863 & -1.836 \\ -0.667 & 0 & 0.667 & 0 \\ -0.091 & -1.445 & 0.091 & 1.445 \end{bmatrix}.$$

The output of the feedforward layer is

$$\boldsymbol{H} = \text{relu}(\boldsymbol{W}_{FF}\boldsymbol{Z}^\top)^\top = \text{relu}\left(\begin{bmatrix} -1.836 & 1.863 & 1.836 & -1.863 \\ 0 & 0.667 & 0 & -0.667 \\ 1.445 & 0.091 & -1.445 & -0.091 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 0 & 1.863 & 1.836 & 0 \\ 0 & 0.667 & 0 & 0 \\ 1.445 & 0.091 & 0 & 0 \end{bmatrix}.$$

Incorporating the residual connection, we finally get

$$\boldsymbol{R} = \boldsymbol{X} + \boldsymbol{H} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1.863 & 1.836 & 0 \\ 0 & 0.667 & 0 & 0 \\ 1.445 & 0.091 & 0 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 2.863 & 1.836 & 0 \\ 0 & 0.667 & 0 & 0 \\ 1.445 & 0.091 & 0 & 1 \end{bmatrix}.$$

2. (5 points) Show that adapter is computing a linear transformation of the input, $\boldsymbol{Z}' = \boldsymbol{C}\boldsymbol{Z}$, and determine the rank of $\boldsymbol{C}$.

   Solution: We have $\boldsymbol{Z}' = \boldsymbol{B}\boldsymbol{A}\boldsymbol{Z} = \underbrace{\boldsymbol{a}\boldsymbol{a}^\top}_{=\boldsymbol{C}}\boldsymbol{Z}$. The rank of $\boldsymbol{C}$ is 1.

3. (10 points) Compute the output of the network **with** the adapter. Note: assume that

$$\boldsymbol{Z} = \begin{bmatrix} -1.863 & 1.836 & 1.863 & -1.836 \\ -0.667 & 0 & 0.667 & 0 \\ -0.091 & -1.445 & 0.091 & 1.445 \end{bmatrix}.$$

   Solution: With the adapter layer, we have $\boldsymbol{H} = \text{relu}((\boldsymbol{W} + \boldsymbol{C})\boldsymbol{Z}^\top)$, with $\boldsymbol{C} = \boldsymbol{a}\boldsymbol{a}^\top$, which since $\boldsymbol{a} = [1, 1, 1, 1]^\top$ is a matrix of ones. Since each row of $\boldsymbol{Z}$ sums to zero, we have that $\boldsymbol{C}\boldsymbol{Z}^\top$ is a matrix of zeros, therefore the adapter has no effect and we obtain the same $\boldsymbol{H}$ and $\boldsymbol{R}$ as in the previous question.